

# 第 14 章

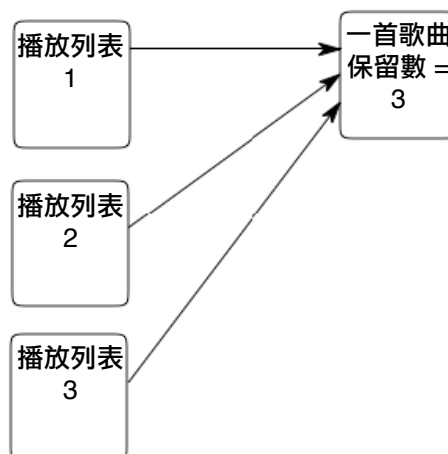
## 記憶體管理

在許多章節中我都為了沒有解釋範例中的某些述句而道歉。那些述句是用來處理記憶體的。您的程式並不是您 Mac 中唯一的程式，而 RAM 是一個珍貴的資源，因此如果您的程式不再需要某部份的記憶體，您應該把它歸還給系統。當您的母親大人告誡您要有禮貌並且與社區鄰居融洽相處時，她正是在教您怎麼程式設計！即使您的程式是唯一在執行的程式，不釋放（free）記憶體最終仍然會將您的程式逼入絕境，而且您的電腦也會慢成龜速。

當您的程式建立一個物件時，這個物件就在記憶體中佔用了一些空間，而您必須在不再使用這個物件之後釋放這塊空間。它的意思是，當您的物件不再使用之後，您應該消滅（destroy）它。然而，要判斷一個物件什麼時候之後就不會再被使用可能並非易事。舉例來說，在程式的執行期間，您的物件可能會被許多其他的物件所參照，因此只要還有其他物件可能會使用到它，它就不能被消滅（試著去使用一個已經被消滅的物件可能會導致您的程式崩潰或產生無法預期的行為）。

為了協助您在物件已經不再需要的時候消滅它們，Cocoa 把一個計數器加入每一個物件中，它代表了物件所謂的 "保留數"（retain count）。在您的程式中，當您儲存了一個對某物件的參照時，您必須把這個物件的保留數加一來讓它知道這件事；相對的，當您移除了一個對某物件的參照時，您就必須把這個物件的保留數減一來知會它。當一個物件的保留數等於零時，這個物件就知道它已經不再由任何地方被參照到並且可以安全的被消滅。接著這個物件就會消滅它自己，並釋放相關的記憶體。

舉例來說，假設您的程式是一個數位點唱機而您有分別用來表示歌曲與播放列表的物件。假設現在有某個歌曲物件被三個播放列表物件參照到，如果它沒有再被其他的地方參照到，您的歌曲物件就會有個等於三的保留數。



一個物件知道它被參照了幾次，而這都要感謝它的保留數。

要增加一個物件的保留數，您只需要傳送一個 *retain* 訊號給這個物件即可；要減少一個物件的保留數，您也只需要傳送一個 *release* 訊息給這個物件。Cocoa 也提供了一個稱為 "autorelease pool"（自動釋放池）的機制，讓您傳送一個延遲的 *release* 訊息給一個物件 - 不是立即，而是稍後。要使用它，您只需要把這個物件註冊到所謂的 autorelease pool 中，這藉由傳送一個 *autorelease* 訊息給您的物件來達成。autorelease pool 會負責傳送這個延遲的 *release* 訊息給您的物件。那些我們在之前的程式中看到與 autorelease pool 有關的述句就是我們為了要正確的設定 autorelease pool 機制而給予系統的指令。

這個為 Cocoa 所使用，也是我們在本章所介紹的記憶體管理技術一般被稱為 *reference counting*（參照計數）。您會在更進階的書或文章中（參見第 15 章）找到 Cocoa 記憶體管理系統的完整說明。一些 Apple 工程師最近暗示 Apple 正在為 Cocoa 製作一個新的記憶體管理模型，稱為 *automatic garbage collection*（自動垃圾收集）。這個模型比目前的更強大、易用而不易出錯。然而在撰寫本書時，Apple 並沒有對何時或者是否會完成並提供這項新科技做出任何保證。

譯註：Apple 已經在 2006 年的 WWDC（World Wide Developer Conference，世界開發者大會）上宣佈會在 2007 年初發佈 Xcode 3.0 時一併發表採用了 automatic garbage collection 技術的 Objective-C 2.0。